Development of an hp-Version Finite Element Method for Computational Optimal Control

Semi-Annual Progress Report
NASA Grant NAG-1-1435
Feb. 22, 1993 – Aug. 21 1993

Prof. Dewey H. Hodges, Principal Investigator
Michael S. Warner, Graduate Research Assistant
School of Aerospace Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332-0150

Research Supported by:
Spacecraft Control Branch
Technical Monitor: Dr. Daniel D. Moerder
Mail Stop 161
NASA Langley Research Center
Hampton, Virginia 23681

Introduction

The purpose of this research effort is to develop a means to use, and to ultimately implement, *hp*-version finite elements in the numerical solution of optimal control problems. Under NAG-939, the hybrid MACSYMA/FORTRAN code GENCODE was developed which utilized *h*-version finite elements to successfully approximate solutions to a wide class of optimal control problems. In that code the means for improvement of the solution was the refinement of the time-discretization mesh. With the extension to *hp*-version finite elements, the degrees of freedom include both nodal values and extra interior values associated with the unknown states, co-states, and controls, the number of which depends on the order of the shape functions in each element. For details, see [1].

Progress During This Period

Optimal Control Problems

A FORTRAN code has been developed using higher-order finite-elements to approximate solutions to a particular subset of optimal control problems. The cost function to be minimized for these problems can contain both an scalar penalty, ϕ, on the states, x, at the initial time or final time, t_f, plus an integral penalty, L, on the states and controls, u, in this form:

$$J = \phi[x(t_0), x(t_f), t_f] + \int_0^{t_f} L(x,u)\ dt \tag{1}$$

The state rates are governed by differential equations

$$\dot{x} = f(x,u) \qquad x \in R^{n_x}, u \in R^{n_u} \tag{2}$$

where f is an autonomous function of the state vector x. The boundary conditions can be specified at the initial time, the final time, or some combination of both, in the form

$$\Psi[x(0), x(t_f), t_f] = 0 \qquad \Psi \in R^{n_{bc}} \tag{3}$$

This formulation also allows for control inequality constraints of the form:

$$g(x,u) \leq 0 \qquad g \in R^{n_p} \qquad n_p < n_u \tag{4}$$

which are enforced through use of slack variables, k.

Adjoining the differential equations, boundary conditions, and control constraints to the original cost function by means of Lagrange multipliers λ, ν, and μ respectively yields a new cost function J':

$$\begin{aligned} J' = {} & \phi[x(t_0), x(t_f), t_f] + \nu^T \Psi[x(0), x(t_f), t_f] + \alpha^T(\hat{x} - x)|_{t_f}^{t_0} + \\ & \int_{t_0}^{t_f} \left\{ L(x,u,t) + \lambda^T[f(x,u) - \dot{x}] + \mu^T[g(x,u) + k^2] \right\} dt \end{aligned} \tag{5}$$

where $\hat{x}$ are the values of the states at the final time, and α is a set of Lagrange multipliers used to ensure that the states are continuous at the initial and final times.

Next, to simplify notation, we define a Hamiltonian as

$$H = L + \lambda^T f + \mu^T g \tag{6}$$

and define $x_f \equiv x(t_f)$ and $x_0 \equiv x(t_0)$ To satisfy the first-order necessary conditions for a local minimum, we follow the development in [1] and take the first variation of J, allowing variations in the states, state rates, controls, Lagrange multipliers, slack variables, and final time, yielding

$$\begin{aligned}
\delta J' = & \left[\frac{\partial \phi}{\partial t} + \nu^T \frac{\partial \Psi}{\partial t} + L + \lambda^T (f - \dot{x}) + \mu^T (g + k^2)\right]_{t_f} dt_f + \delta \nu^T \Psi \\
& + \left(\frac{\partial \phi}{\partial x_f} + \nu^T \frac{\partial \Psi}{\partial x_f}\right) dx(t_f) + \left(\frac{\partial \phi}{\partial x_0} + \nu^T \frac{\partial \Psi}{\partial x_0}\right) dx(t_0) + \delta\alpha(\hat{x} - x)|_{t_0}^{t_f} \\
& + \alpha(d\hat{x} - dx)|_{t_0}^{t_f} + \int_{t_0}^{t_f} \left[\frac{\partial H}{\partial x}\delta x + \frac{\partial H}{\partial u}\delta u - \lambda^T \delta \dot{x} + \delta\lambda^T (f - \dot{x})\right. \\
& \left. + \delta\mu^T (g + k^2) + 2\delta\mu^T k \delta k\right] \, dt
\end{aligned} \tag{7}$$

Given that the variations are continuous, $d\hat{x} = dx$. We then choose $\delta\alpha = d\lambda$, integrate $\lambda^T \delta\dot{x}$ by parts and expand the total differentials at the end points.

$$\begin{aligned}
dx(t_0) &= \delta x(t_0) \\
dx(t_f) &= \delta x(t_f) + \dot{x}(t_f) dt_f \\
d\lambda(t_f) &= \delta\lambda(t_f) + \dot{\lambda}(t_f) dt_f
\end{aligned} \tag{8}$$

Eq. (7) then becomes

$$\begin{aligned}
\delta J' = & \left(\frac{\partial \phi}{\partial x} + \nu^T \frac{\partial \Psi}{\partial x}\right)^T \delta x(t_f) + \left(\frac{\partial \phi}{\partial x_0} + \nu^T \frac{\partial \Psi}{\partial x_0}\right) \delta x(t_0) \\
& + dt_f \left[\dot{x}^T \left(\frac{\partial \phi}{\partial x_f} + \nu^T \frac{\partial \Psi}{\partial x_f}\right) + \dot{\lambda}^T (x - \hat{x}) - \lambda^T \dot{x} + \mu^T k^2\right]_{t_f} \\
& + dt_f \left(\frac{\partial \phi}{\partial t} + \nu^T \frac{\partial \Psi}{\partial t} + H\right)_{t_f} + \delta\nu^T \Psi + \delta\lambda^T (x - \hat{x})|_{t_0}^{t_f} - \lambda^T \delta x|_{t_0}^{t_f} \\
& + \int_{t_0}^{t_f} \left[\frac{\partial H}{\partial x}\delta x + \frac{\partial H}{\partial u}\delta u + \dot{\lambda}^T \delta x + \delta\lambda^T (f - \dot{x})\right. \\
& \left. + \delta\mu^T (g + k^2) + 2\mu^T k \delta k\right] \, dt
\end{aligned} \tag{9}$$

Defining subscripts on H to denote partial derivatives and rearranging terms gives:

$$\begin{aligned}\delta J' = {} & dt_f \left(\frac{\partial \phi}{\partial t} + \nu^T \frac{\partial \Psi}{\partial t} + H\right)_{t_f} + dt_f \dot{x}^T \left(\frac{\partial \phi}{\partial x_f} + \nu^T \frac{\partial \Psi}{\partial x_f} - \lambda\right)_{t_f} \\ & + \left(\frac{\partial \phi}{\partial x_f} + \nu^T \frac{\partial \Psi}{\partial x_f} - \lambda^T\right)_{t_f} \delta x(t_f) + \left(\frac{\partial \phi}{\partial x_0} + \nu^T \frac{\partial \Psi}{\partial x_0} + \lambda^T\right) \delta x(t_0) \\ & + dt_f \dot{\lambda}^T (x - \hat{x}) + dt_f \mu^T k^2|_{t_f} + \delta \nu^T \Psi + \delta \lambda^T (x - \hat{x})|_{t_0}^{t_f} + \int_{t_0}^{t_f} [H_u \delta u \\ & + H_x \delta x + \dot{\lambda}^T \delta x + \delta \lambda^T (f - \dot{x}) + \delta \mu^T (g + k^2) + 2\mu^T k \delta k] \; dt\end{aligned} \tag{10}$$

Now, defining $\hat{\lambda}$ as

$$\hat{\lambda}^T \equiv \begin{cases} -\dfrac{\partial \phi}{\partial x_0} - \nu^T \dfrac{\partial \Psi}{\partial x_0} & t = t_0 \\ \dfrac{\partial \phi}{\partial x_f} + \nu^T \dfrac{\partial \Psi}{\partial x_f} & t = t_f \end{cases} \tag{11}$$

and integrating the terms $\dot{\lambda}^T \delta x$ and $\delta \lambda^T \dot{x}$ by parts, Eq. (10) becomes:

$$\begin{aligned}\delta J' = {} & dt_f \left(\frac{\partial \phi}{\partial t} + \nu^T \frac{\partial \Psi}{\partial t} + H\right)_{t_f} + \lambda^T \delta x|_{t_0}^{t_f} - \delta \lambda^T x|_{t_0}^{t_f} \\ & + \left[\left(\hat{\lambda} - \lambda\right)^T \delta x\right]_{t_0}^{t_f} + \delta \nu^T \Psi + dt_f \left[\dot{x}^T (\hat{\lambda} - \lambda)\right]_{t_f} \\ & - \left[(\hat{x} - x)^T \delta \lambda\right]_{t_0}^{t_f} + dt_f \mu^T k^2|_{t_f} - dt_f \left[\dot{\lambda}^T (\hat{x} - x)\right]_{t_f} \\ & + \int_{t_0}^{t_f} \left[H_u \delta u + H_x \delta x - \lambda^T \delta \dot{x} + \delta \lambda^T f + \delta \dot{\lambda}^T x \right. \\ & \left. + \delta \mu^T (g + k^2) + 2\mu^T k \delta k\right] dt\end{aligned} \tag{12}$$

We will now enforce that $\hat{x} = x$ and $\hat{\lambda} = \lambda$ at t_0 and t_f and define

$$\hat{H} \equiv \frac{\partial \phi}{\partial t} + \nu^T \frac{\partial \Psi}{\partial t} \tag{13}$$

then Eq. (12) becomes

$$\begin{aligned}\delta J' = {} & dt_f \left(\hat{H} + H\right)_{t_f} + dt_f \mu^T k^2|_{t_f} + \delta \nu^T \Psi + \hat{\lambda}^T \delta x|_{t_0}^{t_f} - \hat{x}^T \delta \lambda|_{t_0}^{t_f} \\ & + \int_{t_0}^{t_f} \left[H_u \delta u + H_x \delta x - \delta \dot{x}^T \lambda + \delta \lambda^T f + \delta \dot{\lambda}^T x \right. \\ & \left. + \delta \mu^T (g + k^2) + 2\mu^T k \delta k\right] dt\end{aligned} \tag{14}$$

The time interval is broken up into N not necessarily equal length time elements, Δt_i, such that the time at each element boundary $\hat{t}_i$ is calculated as:

$$\begin{aligned} \hat{t}_1 &= t_0 \\ \hat{t}_i &= \hat{t}_{i-1} + \Delta t_i \qquad i = 2, \ldots, N+1 \end{aligned} \tag{15}$$

and we define the states and controls at these nodes to be

$$\hat{x}_i = x(\hat{t}_i) \qquad i = 1, \ldots, N+1 \tag{16}$$

$$\hat{u}_i = u(\hat{t}_i) \qquad i = 1, \ldots, N+1 \tag{17}$$

The time within the i^{th} element, t_i is expressed as $t_i = \hat{t}_{i-1} + \tau \Delta t_i$ where $0 \leq \tau \leq 1$ and

$$\tau = \frac{t_i - \hat{t}_{i-1}}{\Delta t_i} \tag{18}$$

so that $dt = \Delta t_i d\tau$. Substituting these relationships into Eq. (14) gives:

$$\begin{aligned} \delta J' = {} & dt_f \left(\hat{H} + H \right)_{t_f} + dt_f \mu^T k^2|_{t_f} + \delta \nu^T \Psi + \hat{\lambda}^T \delta x|_{t_0}^{t_f} - \hat{x}^T \delta \lambda|_{t_0}^{t_f} \\ & + \sum_i^N \Delta t_i \int_0^1 \left[H_{u_i} \delta u_i + H_{x_i} \delta x_i + \frac{\delta \lambda_i'^T}{\Delta t_i} x_i - \frac{\delta x_i'^T}{\Delta t_i} \lambda_i \right. \\ & \left. + \delta \lambda_i^T f_i + \delta \mu_i^T (g_i + k_i^2) + 2 \mu_i^T k_i \delta k_i \right] d\tau \end{aligned} \tag{19}$$

where subscript i refers either to quantities within the i^{th} element or to functions evaluated using quantities within the i^{th} element.

Higher-Order Shape Functions

From [2], we define C^0 shape functions for the variations of state equations' Lagrange multipliers (or *costates*) and states in each element in terms of nodal values (superscript $\hat{}$) and internal values (superscript $\bar{}$).

$$\begin{aligned} \delta \lambda_i &= \delta \hat{\lambda}_i (1 - \tau) + \delta \hat{\lambda}_{i+1} \tau + \sum_{j=1}^{n_b - 1} (1 - \tau) \tau \beta_j(\tau) \delta \bar{\lambda}_{ij} \\ \delta x_i &= \delta \hat{x}_i (1 - \tau) + \delta \hat{x}_{i+1} \tau + \sum_{j=1}^{n_b - 1} (1 - \tau) \tau \beta_j(\tau) \delta \bar{x}_{ij} \end{aligned} \tag{20}$$

Here n_b is the order of the shape function polynomial being used, with $n_b = 1$ representing h-version shape functions, and the summation would be ignored. The functions $\beta_j(\tau)$ are polynomials of order $(j-1)$ as defined in [2]. That functional form is necessitated by the

time derivatives in Eq. (19) and required end conditions. The time derivative expression looks like

$$\frac{d\delta x_i}{d\tau} \equiv \delta x_i' = \Delta t_i \delta \dot{x}_i = -\delta \hat{x}_i + \delta \hat{x}_{i+1} + \sum_{j=1}^{n_b-1} \gamma_j(\tau) \delta \bar{x}_{ij} \tag{21}$$

where

$$\gamma_j(\tau) \equiv [(1-\tau)\tau\beta_j(\tau)]'$$

Similar expressions hold for the costates.

No time derivatives of the states and costates themselves are needed in Eq. (19), so simpler shape functions are used

$$x_i = \begin{cases} \hat{x}_i & \tau = 0 \\ \sum_{j=1}^{n_b} \alpha_j(\tau) \bar{x}_{ij} & 0 < \tau < 1 \\ \hat{x}_{i+1} & \tau = 1 \end{cases} \qquad \lambda_i = \begin{cases} \hat{\lambda}_i & \tau = 0 \\ \sum_{j=1}^{n_b} \alpha_j(\tau) \bar{\lambda}_{ij} & 0 < \tau < 1 \\ \hat{\lambda}_{i+1} & \tau = 1 \end{cases} \tag{22}$$

where the functions $\alpha_j(\tau)$ are again polynomials of order $j-1$ as defined in [2].

For the controls, control constraint Lagrange multipliers, slack variables, and their variations, again no time derivatives exist in Eq. (19), so the same easier shape functions are used:

$$\delta u_i = \sum_{j=1}^{n_b} \alpha_j(\tau) \delta \bar{u}_{ij} \tag{23}$$

$$\delta \mu_i = \sum_{j=1}^{n_b} \alpha_j(\tau) \delta \bar{\mu}_{ij} \qquad \delta k_i = \sum_{j=1}^{n_b} \alpha_j(\tau) \delta \bar{k}_{ij} \tag{24}$$

$$u_i = \sum_{j=1}^{n_b} \alpha_j(\tau) \bar{u}_{ij} \qquad 0 < \tau < 1 \tag{25}$$

$$\mu_i = \sum_{j=1}^{n_b} \alpha_j(\tau) \bar{\mu}_{ij} \qquad 0 < \tau < 1 \qquad k_i = \sum_{j=1}^{n_b} \alpha_j(\tau) \bar{k}_{ij} \qquad 0 < \tau < 1 \tag{26}$$

Substituting these relationships into Eq.(19) gives

$$\begin{aligned}
\delta J' = {} & dt_f \hat{\mu}_{N+1}^T \hat{k}_{N+1}^2 + \hat{\lambda}_{N+1}^T \delta x_{N+1} - \hat{\lambda}_1^T \delta x_1 - \hat{x}_{N+1}^T \delta \lambda_{N+1} + \hat{x}_1^T \delta \lambda_1 \\
& + \delta \nu^T \Psi + dt_f \left(\hat{H} + H \right)_{t_f} + \sum_i^N \langle \Delta t_i \int_0^1 \left\{ H_{u_i} \left(\sum_{j=1}^{n_b} \alpha_j(\tau) \delta \bar{u}_{ij} \right) \right. \\
& + H_{x_i} \left[\delta \hat{x}_i (1-\tau) + \delta \hat{x}_{i+1} \tau + \sum_{j=1}^{n_b - 1} \epsilon_j(\tau) \delta \bar{x}_{ij} \right] \\
& + f_i^T \left[\delta \hat{\lambda}_i (1-\tau) + \delta \hat{\lambda}_{i+1} \tau + \sum_{j=1}^{n_b - 1} \epsilon_j(\tau) \delta \bar{\lambda}_{ij} \right] \\
& + \left[\sum_{j=1}^{n_b} \alpha_j(\tau) \delta \bar{\mu}_{ij} \right]^T \left[g_i + \sum_{j=1}^{n_b} \alpha_j(\tau) \bar{k}_{ij}^2 \right] \\
& \left. + 2 \left(\sum_{j=1}^{n_b} \alpha_j(\tau) \bar{\mu}_{ij} \right)^T \left(\sum_{j=1}^{n_b} \alpha_j(\tau) \bar{k}_{ij} \right) \sum_{j=1}^{n_b} \alpha_j(\tau) \delta \bar{k}_{ij} \right\} d\tau \rangle \\
& + \sum_i^N \int_0^1 \left[-\delta \hat{\lambda}_i + \delta \hat{\lambda}_{i+1} + \sum_{j=1}^{n_b - 1} \gamma_j(\tau) \delta \bar{\lambda}_{ij} \right]^T \sum_{j=1}^{n_b} \alpha_j(\tau) \bar{x}_{ij} \, d\tau \\
& - \sum_i^N \int_0^1 \left[-\delta \hat{x}_i + \delta \hat{x}_{i+1} + \sum_{j=1}^{n_b - 1} \gamma_j(\tau) \delta \bar{x}_{ij} \right]^T \sum_{j=1}^{n_b} \alpha_j(\tau) \bar{\lambda}_{ij} \, d\tau
\end{aligned} \tag{27}$$

where

$$\epsilon_j(\tau) \equiv (1-\tau)\tau\beta_j(\tau)$$

Due to the orthogonality of the chosen polynomials $\alpha(\tau)$ and $\gamma(\tau)$, this equation reduces to

$$
\begin{aligned}
\delta J' = {} & dt_f \hat{\mu}_{N+1}^T \hat{k}_{N+1}^2 + \hat{\lambda}_{N+1}^T \delta x_{N+1} - \hat{\lambda}_1^T \delta x_1 - \hat{x}_{N+1}^T \delta \lambda_{N+1} + \hat{x}_1^T \delta \lambda_1 \\
& + \delta \nu^T \Psi + dt_f \left(\hat{H} + H \right)_{t_f} + \sum_i^N \Delta t_i \int_0^1 \left\{ H_{u_i} \left(\sum_{j=1}^{n_b} \alpha_j(\tau) \delta \bar{u}_{ij} \right) \right. \\
& + \delta \hat{x}_i^T (1-\tau) H_{x_i} + \delta \hat{x}_{i+1}^T \tau H_{x_i} + \delta \hat{\lambda}_i^T (1-\tau) f_i + \delta \hat{\lambda}_{i+1}^T \tau f_i \\
& + \sum_{j=1}^{n_b - 1} \left[\epsilon_j(\tau) \delta \bar{x}_{ij}^T H_{x_i} + \epsilon_j(\tau) \delta \bar{\lambda}_{ij}^T f_i \right] \\
& + \left(\sum_{j=1}^{n_b} \alpha_j(\tau) \delta \bar{\mu}_{ij} \right)^T \left[g_i + \sum_{j=1}^{n_b} \alpha_j(\tau) \bar{k}_{ij}^2 \right] \\
& \left. + 2 \left(\sum_{j=1}^{n_b} \alpha_j(\tau) \bar{\mu}_{ij} \right)^T \left(\sum_{j=1}^{n_b} \alpha_j(\tau) \bar{k}_{ij} \right) \sum_{j=1}^{n_b} \alpha_j(\tau) \delta \bar{k}_{ij} \right\} d\tau \\
& + \sum_i^N \left(-\delta \hat{\lambda}_i^T \bar{x}_{i,1} + \delta \hat{\lambda}_{i+1}^T \bar{x}_{i,1} - \sum_{j=2}^{n_b} \delta \bar{\lambda}_{i,j-1}^T \bar{x}_{ij} \right) \\
& + \sum_i^N \left(\delta \hat{x}_i^T \bar{\lambda}_{i,1} - \delta \hat{x}_{i+1}^T \bar{\lambda}_{i,1} + \sum_{j=2}^{n_b} \delta \bar{x}_{i,j-1} \bar{\lambda}_{ij} \right)
\end{aligned}
\tag{28}
$$

Rearranging terms gives

$$
\begin{aligned}
\delta J' = {} & \delta\hat{\lambda}_1^T \left[-\hat{x}_1 + \bar{x}_{1,1} - \Delta t_1 \int_0^1 (1-\tau) f_1 \, d\tau \right] \\
& + \sum_{i=2}^{N} \delta\hat{\lambda}_i^T \left[-\bar{x}_{i-1,1} - \Delta t_{i-1} \int_0^1 \tau f_{i-1} \, d\tau + \bar{x}_{i,1} - \Delta t_i \int_0^1 (1-\tau) f_i \, d\tau \right] \\
& + \delta\hat{\lambda}_{N+1}^T \left[-\bar{x}_{N,1} - \Delta t_N \int_0^1 \tau f_N \, d\tau + \hat{x}_{N+1} \right] \\
& + \sum_{i=1}^{N} \sum_{j=1}^{n_b - 1} \delta\bar{\lambda}_{ij}^T \left[-\bar{x}_{i,j+1} + \Delta t_i \int_0^1 \epsilon_j(\tau) f_i \, d\tau \right] \\
& + \delta\hat{x}_1^T \left\{ \hat{\lambda}_1 - \bar{\lambda}_{1,1} - \Delta t_i \int_0^1 (1-\tau) \left[L_{x_1} + \sum_{j=1}^{n_b} \alpha_j(\tau) \bar{\lambda}_{1,j}^T f_{x_1} \right] d\tau \right\} \\
& + \sum_{i=2}^{N} \delta\hat{x}_i^T \left\{ -\bar{\lambda}_{i,1} - \Delta t_i \int_0^1 (1-\tau) \left[L_{x_i} + \sum_{j=1}^{n_b} \alpha_j(\tau) \bar{\lambda}_{ij}^T f_{x_i} \right] d\tau \right. \\
& \qquad \left. + \bar{\lambda}_{i-1,1} - \Delta t_{i-1} \int_0^1 \tau \left[L_{x_{i-1}} + \sum_{j=1}^{n_b} \alpha_j(\tau) \bar{\lambda}_{i-1,j}^T f_{x_{i-1}} \right] d\tau \right\} \qquad (29) \\
& + \sum_{i=1}^{N} \sum_{j=1}^{n_b - 1} \delta\bar{x}_{ij}^T \left\{ \bar{\lambda}_{i,j+1} + \Delta t_i \int_0^1 \epsilon_j(\tau) \left[L_{x_i} + \sum_{j=1}^{n_b} \alpha_j(\tau) \bar{\lambda}_{ij}^T f_{x_i} \right] d\tau \right\} \\
& + \delta\hat{x}_{N+1}^T \left\{ -\hat{\lambda}_{N+1} + \bar{\lambda}_{N,1} - \Delta t_N \int_0^1 \tau \left[L_{x_N} + \sum_{j=1}^{n_b} \alpha_j(\tau) \bar{\lambda}_{N,j}^T f_{x_N} \right] d\tau \right\} \\
& + dt_f \hat{\mu}_{N+1}^T \hat{k}_{N+1}^2 + \delta\nu^T \Psi \\
& + dt_f \left(\hat{H} + H \right)_{t_f} + \sum_{i}^{N} \sum_{j=1}^{n_b} \delta\bar{u}_{ij} \Delta t_i \int_0^1 H_{u_i} \alpha_j(\tau) \cdot d\tau \\
& + \sum_{i}^{N} \sum_{j=1}^{n_b} \delta\bar{\mu}_{ij} \Delta t_i \int_0^1 \alpha_j(\tau) \left[g_i + \sum_{j=1}^{n_b} \alpha_j(\tau) \bar{k}_{ij}^2 \right] d\tau \\
& + \sum_{i}^{N} \sum_{j=1}^{n_b} \delta\bar{k}_{ij} \Delta t_i \int_0^1 2\alpha_j(\tau) \left[\sum_{j=1}^{n_b} \alpha_j(\tau) \bar{\mu}_{ij} \right]^T \left[\sum_{j=1}^{n_b} \alpha_j(\tau) \bar{k}_{ij} \right] d\tau
\end{aligned}
$$

This grouping of terms shows the equations to be solved as coefficients of the various variational quantities. When the coefficients are set equal to zero, the variation of J' is

zero, approximating the first-order necessary conditions for optimal control. In addition to the above, the costate boundary conditions need to be enforced, as provided for previously

$$\begin{aligned} \hat{\lambda}_1^T + \frac{\partial \phi}{\partial x_0} + \nu^T \frac{\partial \Psi}{\partial x_0} &= 0 \\ \hat{\lambda}_{N+1}^T - \frac{\partial \phi}{\partial x_f} - \nu^T \frac{\partial \Psi}{\partial x_f} &= 0 \end{aligned} \tag{30}$$

The equations when there are multiple phases are similar and can also be handled by the code. Extra boundary conditions have to be specified, and the appropriate jump conditions for the costates and Hamiltonian are handled automatically.

Implementation

The above equations are solved using a restricted-step Newton-Raphson method, as implemented in a FORTRAN code. Sparse linear systems solvers from the Harwell subroutine library [3] are used. The user needs to specify an initial guess for all of the variables for the Newton-Rapheson iteration.

The symbolic-manipulation package Macsyma developed by Symbolics [4] is used to generate analytic partial derivatives. Macsyma is also used to generate the $\alpha(\tau)$ and $\epsilon(\tau)$ polynomials, which come from a recursion formula involving derivatives and integrals of polynomials, as developed in [2]. The user specifies the order of the polynomials, and Macsyma generates the necessary ones. At this time all variables have the same order shape function, but eventually the order for each variable will be able to be independently specified. The order of the shape functions can be changed between runs, but only if the desired number is less than that specified when the Macsyma-generated code was made. Otherwise the necessary polynomials will not be available.

The integrals in Eq. 29 are approximated using Gaussian integration, with the user selecting the number of Gauss points, which at this time is constant for all of the integrations. That number, as well as the number of elements can be adjusted between runs through use of a namelist file and appropriate changes to the initial guess file.

Results

The code has been tested on a linear, single-state, single-control system and on a multiple-state, single-control system with nonlinear system dynamics.

Linear Problem The linear problem we considered is a minimum energy problem for getting from one position to another. The system dynamics are

$$\dot{x} = x + u \tag{31}$$

where x is the state and u is the control. The boundary conditions are

$$\begin{aligned} \Psi_1 &= x_0 - (e-1) \\ \Psi_2 &= x_f - (1-e) = 0 \end{aligned} \tag{32}$$

and the cost function is

$$J = \int_0^1 \frac{1}{2} u^2 \, dt \tag{33}$$

The analytic solution to this problem is

$$\begin{aligned} x(t) &= e^{(1-t)} - e^t \\ \lambda(t) &= 2e^{(1-t)} \\ u(t) &= -2e^{(1-t)} \end{aligned} \tag{34}$$

This problem was run for a variety of shape function orders, numbers of Gauss points, and numbers of elements. The code assumes a free time problem, introducing a nonlinear equation $(H(t_f) - \hat{H} = 0)$ even into linear problems. Knowing this, in all cases all of the linear equations were solved in a single iteration, while that last nonlinear equation converged better than quadratically. This problem is easy enough to work by hand reasonably, and the Jacobian matrix generated by the code matched the one we derived explicitly for the second-order shape function case.

Also, since all of the errors in all of the equations in (29) were solved to within $1.0e-10$, the values for the costate were always just the negative of those for the control and the state boundary conditions were met. Therefore in comparing results, we will look at the initial and final values of only the control.

Table 1: Errors in initial and final values of control for various values of higher-order finite-element parameters

$u(t_0)$ Error	$u(t_f)$ Error	Gauss Points	Shape Fn. Order	Number of Elements
5.18e-2	1.41e-1	1	1	1
8.58e-4	2.33e-3	2	2	1
6.00e-6	1.63e-5	3	3	1
2.36e-8	6.41e-8	4	4	1
5.92e-11	1.61e-10	5	5	1
1.23e-2	3.34e-2	2	1	2
5.12e-5	1.39e-4	2	2	2
9.12e-8	2.48e-7	3	3	2
9.01e-11	2.45e-10	4	4	2
5.35e-14	1.50e-13	5	5	2

As one can see, the errors reduce dramatically as the order of the shape functions increases. Not shown is a similar decrease in error as the number of Gauss points increases, to a minimum when the number of Gauss points equals the order of the element. This is similar performance to the time-marching algorithm, as presented in our previous report. With further analysis we will do trade studies comparing the computational effort necessary to solve the problem with more elements vs. using higher-order shape functions.

Nonlinear Problem Next we computed some preliminary solutions to a problem with nonlinear system dynamics that involves maximum velocity transfer of a particle of mass m to a rectilinear path in a fixed time (see [5], pg. 59). The mass is acted on by a force of constant magnitude ma and variable heading $\beta(t)$. The states for this problem are position of the particle x (horizontal) and y (vertical) and the corresponding velocity components u and v. The differential equations for this system are then:

$$\begin{aligned} \dot{x} &= u \\ \dot{y} &= v \\ \dot{u} &= a\cos\beta \\ \dot{v} &= a\sin\beta \end{aligned} \tag{35}$$

with initial conditions corresponding to a zero velocity at the origin and with terminal constraints that the particle is in horizontal flight at a given height (assumed here to be 1)

$$\begin{aligned} \Psi_1 &= u(0) = 0 \\ \Psi_2 &= v(0) = 0 \\ \Psi_3 &= x(0) = 0 \\ \Psi_4 &= y(0) = 0 \\ \Psi_5 &= y(t_f) - 1 = 0 \\ \Psi_6 &= v(t_f) = 0 \end{aligned} \tag{36}$$

for an unspecified final horizontal position $x(t_f)$ and for a final horizontal velocity $u(t_f)$ to be maximized. The cost function is then

$$J = u(t_f) \tag{37}$$

Reference [5] gives the analytic solution in terms of the initial force heading angle, the final time, and the final altitude in unspecified units. These values were chosen to be 75°, 1, and 1 respectively. In all cases, all of the boundary conditions were met and the horizontal velocity was twice the horizontal position, leaving the final values of horizontal velocity, horizontal position costate and heading angle as the three most interesting quantities to look at for comparison purposes.

Table 2: Errors in final values of $U(t_f)$, $\lambda_y(t_f)$, and $\beta(t_f)$ for various values of higher-order finite-element parameters

$U(t_f)$ error	$\lambda_y(t_f)$ error	$\beta(t_f)$ error	Number of Elements	Shape Fn. Order
1.90e-01	4.25e-02	8.07e-03	2	1
2.77e-02	7.34e-02	1.39e-02	2	2
N/A	N/A	N/A	3	1
1.89e-03	7.79e-03	1.49e-03	3	2
6.77e-02	7.53e-02	1.42e-02	4	1
2.80e-03	1.55e-02	2.95e-03	4	2
5.31e-02	1.21e-01	2.28e-02	5	1
4.77e-04	1.52e-03	2.90e-04	5	2

Preliminary results are shown in Table 2. Missing values were for cases that did not converge, which is not surprising given the small number of elements being used. For this nonlinear problem the improvement with the higher-order element is not as dramatic. For 4 and 5 elements all errors decrease with the higher-order elements. For 2 elements some errors do not decrease. This may be due to the need for a larger number of Gauss points when the order is increased (6 points were used). For the simplest (h-version) element, 1 Gauss point is optimum in the sense that it is the smallest number of points which gives acceptable error. However, when one adds the higher-order shape functions, a larger number of Gauss points will always be required for nonlinear problems than for linear ones.

The optimum number of Gauss points seems to be dependent on both the maximum element order and the types of nonlinearities in the problem. It is possible that there is a simple calculation, using some terms from the equations under consideration, which can be done for any given problem and which will approximately determine the optimum number of Gauss points for that particular problem as a function of the orders of element from 2 up to the maximum needed. Whether this is in fact true should become clearer as more problems are tried and more cases of these problems are run. Recent developments by Hinnant (of the Computational Mechanics Branch at Langley) may also lead to savings in the numerical quadrature costs.

One possible drawback of higher-order finite element schemes is the increased computational effort within each element required in implementing hp-version finite elements. We will ultimately determine whether this computational effort is sufficiently offset by the reduction in the number of time elements used and improvements in the Newton-Raphson convergence so as to be useful in solving optimal control problems in real time. Also, because certain of the element interior unknowns can be eliminated at the element level by solving a small set of nonlinear algebraic equations in which the nodal values are taken as given, the scheme may turn out to be especially powerful in a parallel computing environment. A different processor could be assigned to each element. The number of processors,

strictly speaking, would not be required to be any larger than the number of sub-regions which are free of discontinuities.

References

[1] Bless, Robert R.: "Time-Domain Finite Elements in Optimal Control with Application to Launch Vehicle Guidance." Ph.D. Dissertation, School of Aerospace Engineering, Georgia Institute of Technology, Mar. 1991 (also NASA CR 4376, Mar. 1991).

[2] Hodges, Dewey H.; and Hou, Lin-Jun: "Shape Functions for Mixed p-version Finite Elements in the Time Domain." *J. Sound and Vibration*, vol. 145, no. 2, Mar. 8, 1991, pp. 169 – 178.

[3] Duff, I. S., Harwell Subroutine Library, Computer Science and Systems Division, Harwell Laboratory, Oxfordshire, England, February 1988, Chapter M.

[4] MACSYMA *Reference Manual*, Symbolics, Inc., Burlington, MA, 1988.

[5] Bryson, Arthur E., and Ho, Yu-Chi, *Applied Optimal Control – Optimization, Estimation, and Control*, Hemisphere Publishing Corporation, New York, NY, 1975.